The necessity of optimal design for parallel machines and a possible certified methodology

J-P. Merlet
INRIA Sophia-Antipolis, France

Abstract: Although they have many advantages in term of positioning accuracy, stiffness, load capacity parallel machines have also a main drawback: their performances are very sensitive to their dimensioning. Hence although the choice of a given mechanical structure among the numerous possibilities that are offered for parallel machines may influence the performances of the machine the rule of thumb is: a mechanically appropriate but poorly dimensioned machine will present in general largely lower performances than a well designed machine with a mechanical architecture a priori less adequate.

Optimal dimensioning of a parallel machine is hence a critical issue but also a complex one, especially if uncertainties in the manufacturing are taken into account. We will present a possible design methodology based on interval analysis and will illustrate this methodology on realistic examples.

1 Introduction

It is not so well known that the performances of parallel robots are highly sensitive to their geometric design. Consider for example a Gough platform whose attachment points located on the base have as coordinates:

$A_1 : (-9, 9)$ $A_2 : (9, 9)$ $A_3 : (12, -3)$ $A_4 : (3, -13)$ $A_5 : (-3, -13)$ $A_6 : (-12, -3)$

The attachment points on the platform have a classical repartition on a circle of radius r_1 with 2 adjacent points separated by an angle of 30 degrees. We then consider the stiffness matrix K of the robot, assuming a unit value for the link stiffness which will lead to $K = J^{-T}J^{-1}$ at a particular pose for the platform (base and platform are parallel and the center of the base and platform are located on the vertical axis). Figure 1 presents the variation of k_{θ_x} as a function of r_1. It may be seen that for a variation of r_1 from 3 to 9 the variation of k_{θ_x} is roughly from 20 to 200. It may also been intuitively understood that such scale factor will get worse when we consider all poses within the workspace of the robot. This example shows clearly the importance of a well tought design process.

Design synthesis is a two-step process:

- *structure synthesis*: determine the general arrangement of the mechanical structure such as the type and number of joints and the way they will be connected
- *dimensional synthesis*: determine the length of the links, the axis and location of the joints, ... In this paper the word dimension will have the

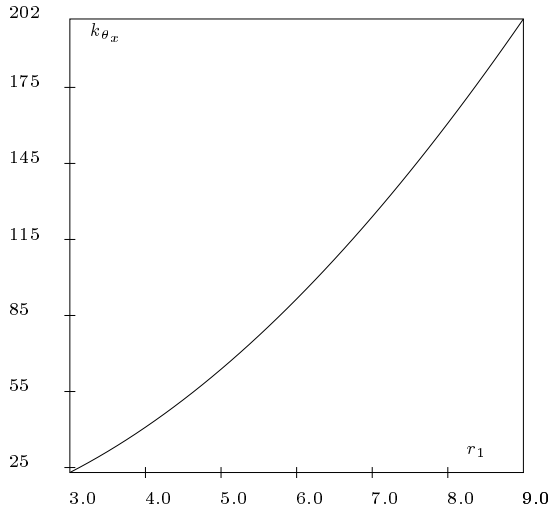

Figure 1: Variation of the stiffness element k_{θ_x} as a function of the platform radius r_1

broad sense of any parameter that will influence the robot behavior and is needed for the manufacturing of the robot

For serial robots general trends for the robot performances may be deduced from the structure. For example we may compare the reachable workspace of 3 d.o.f. robot of type PPP and RRR: assuming a stroke of L for the linear actuator and a length L for the links the PPP workspace volume will be L^3 while it will be $\approx 40L^3$ for the RRR robot. But even for serial robot such trends will not be sufficient to fully determine the optimal robot: indeed many performances have to be taken into account for defining an optimal robot, some of them being highly dependent upon the dimensions of the robot (for example the load capacity). The case is worse for closed-chain robots for which such general trends cannot be easily derived

Hence optimal design for a robot implies both type of synthesis but our experience in the design of closed-chain robots has led us to the following rule of a thumb: *a robot with an a-priori more appropriate mechanical structure but whose dimensions have been poorly chosen will exhibit largely lower performances than a well dimensionally designed robot with an a-priori less appropriate structure.* We are not claiming that structural synthesis is not an important area but that it cannot be disconnected from dimensional synthesis. The point is that structural synthesis, although still in progress, has strong theoretical backgrounds (such as screw and group theories) while, as we will see, dimensional synthesis lack of such background.

2 Dimensional synthesis: state of the art

Dimensional synthesis is a problem that has attracted a lot of attention but most of the works focus on design for a specific robot's feature such as workspace [1, 5, 9, 10, 11] or accuracy [7, 18, 19].

The usual way to solve the optimal design problem is to define a real-valued function C as a weighted sum of performance indices P_i [4] with a value in the range [0,1]. These indices measure how much each of the performance requirement is violated for a given robot. A value equal to 0 indicates that the requirement is fully satisfied while a value 1 is used when the requirement is fully violated. These performance indices are clearly functions of the design parameters set $\mathcal{P}$. The cost function is then defined as

$$C = \sum_i w_i P_i(\mathcal{P})$$

where w_i are weights. It is assumed that the optimal design solution is obtained for the value of the parameters in $\mathcal{P}$ that minimize C and a numerical procedure is used to find the values of $\mathcal{P}$ which minimize C, usually starting with an initial guess $\mathcal{P}_0$. But this method has many drawbacks and we will mention a few of them.

First it is assumed that the requirement indices can be defined and that they can be calculated efficiently (indeed the numerical optimization procedure requires a large number of evaluation of these indices). All these assumptions are difficult to realize in practice for robots: for example what could be the definition of an index that indicates that a cube of given volume must be included in the robot's workspace ? Evaluation of some indices may also be a quite difficult problem: for example we may define as index the worst positioning error along a given axis for any pose of the robot within a prescribed workspace and evaluating this index is by itself a difficult constrained optimization problem.

A second drawback of the cost-function approach is the difficulty in the determination of the weights. These weights are present in the function not only to indicate the priority of the requirements but also to tackle with the units problem in the performance indices. For example if the used performance indices are the workspace volume and positioning accuracy for a 3-dof translational robot we are dealing with quantities whose units differ by a ratio of 10^3: hence the weights must be used to normalize the indices. The choice of the weights is therefore essential while there is not intuitive rules for determining their values. Furthermore a small change in the weights may lead to very different optimal designs.

Even if the cost-function is effective it may lead to inconclusive result. This was exemplified by Stoughton [20] who was wanting to determine special kind of Gough platform with improved dexterity and a reasonable workspace volume. Hence Stoughton has considered two criteria in his cost-function: the dexterity and the workspace volume. He find out that these criteria were varying in opposite ways: the dexterity was decreasing when the workspace volume was

increasing. Hence there was no optimal design solution per se and the problem was in fact to determine an acceptable compromise between the two requirements. This advocates the point that in optimal design we should not try to maximize one performance without imposing constraint on the minimal values of other performances (for example Gosselin [6] shows that the Gough platform having the largest workspace for a given stroke of the actuator will have a geometry such that it cannot be controlled). It may also be considered that in some cases some requirements are *imperative* i.e. they must never be violated while some others may be somewhat relaxed. But imposing an imperative requirements in the cost-function is difficult without violating the differentiability constraint and/or allowing large violation on the other constraints.

A third drawback of the cost-function approach is that it provides only one solution. This causes three main problems:

- *manufacturing tolerances* will be such that the real robot will differ from the theoretical one. Hence with only one theoretical design solution we cannot guarantee that the real robot will fulfills the requirements
- providing only one solution does not allow to consider secondary requirements that may have not been used in the cost-function but may be a decision factor if two robots satisfy in a similar way the main requirements
- for providing only one solution we have to assume that the designer masters all the criterion that will lead the end-user to a solution. This is seldom the case in practice: for example economic considerations will usually play a role although the designer cannot be fully aware of their level of implication

We will propose now another design methodology.

3 Another design methodology: the parameters space approach

We will first define the *parameters space* $\mathcal{S}^n$ as a n-dimensional space in which each dimension corresponds to one of the n design parameters of the robot. Hence a point in that space correspond to one unique design of the robot.

Consider now a list of m requirements $\{\mathcal{R}_1, \ldots, \mathcal{R}_m\}$ that define minimal or maximal allowed values of some robot's performance (such as accuracy, stiffness, ...) or some required properties (for example that a set of pre-defined trajectories lie within the robot's workspace) and assume that we are able to design an algorithm that is able to calculate the region $\mathcal{Z}_i$ defined as the region of the parameter space $\mathcal{S}^n$ that includes all the robot's design that satisfies the requirement $\mathcal{R}_i$. Then the intersection of all the $\mathcal{Z}_i$ will define **all** the robot's design that satisfies **all** the requirements. With this approach we will have found

a **complete** answer to the optimal design problems as we will have determined all possible design solutions.

To make this approach practical we are confronted to two difficulties:

1. calculating the region $\mathcal{Z}_i$
2. computing the intersection of the regions

The calculation of the region is indeed quite difficult as we have basically to determine regions whose borders are determined by a set of complex highly non-linear relations (but in some cases this may be possible if the number of design parameters is not too high, see [13, 16]). But a good point is that it is not necessary to determine these regions *exactly*. Indeed determining points of the region close to the border does not make sense as if they are chosen as nominal parameter value, then the real robot, whose parameter are affected by manufacturing tolerances, may in fact have a representative point in the parameters space that is outside the $\mathcal{Z}_i$ regions. Hence computing an *approximation* of the regions whose border is sufficiently close to the real border is sufficient.

It is now well known that *interval analysis* may be an appropriate tool for computing this approximation. Indeed each of the n design parameters $\mathcal{P}_i$ in $\mathcal{P}$ represents a physical quantity (e.g. link length, rotation axis,...) that can usually be bounded i.e. we can assign a range for each parameter and we are looking only for design solutions such that the values of the parameters lie within their assigned range. Then interval analysis will be able to provide, for example, an approximation of all values of the design parameters such that a set of inequalities $F(\mathcal{P}) \leq 0$. The result will be a list of m elements, each element being constituted of n ranges $\mathcal{R}^j_{\mathcal{P}_i}$, $j \in [1, m], i \in [1, n]$, one for each design parameter. Choosing as value of the design parameters an arbitrary number within the ranges of a given element ensures that the set of inequalities will be satisfied. The quality of the approximation only depend upon the minimal width of the ranges that are allowed in the element. Examples of applications of such algorithm are described in [2] (force transmission in a 3 d.o.f. robot), [8] (workspace requirements), [14] (singularity detection).

4 Optimal design

We have seen that our optimal design approach requires the calculation of the regions $\mathcal{Z}$ and then their intersection. Interval analysis seems to be quite appropriate for the second part. Indeed if we assume that we are able to obtain the regions $\mathcal{Z}$ as a set of boxes, then calculating their intersection is a classical problem in computational geometry that can be solved easily.

We are now confronted to the problem of calculating the region $\mathcal{Z}$ using interval analysis. As mentioned previously there is no need to calculate *exactly* these regions as points on the border cannot be considered as nominal design parameter values because the effect of manufacturing tolerances may put the value of the *real* robot parameter outside the region $\mathcal{Z}$. This point may be used as an advantage for interval analysis-based method by using the following rule:

the result of the algorithm should be a set of boxes such that for each box the range for each design parameter has a width which is at least equal to the manufacturing tolerance for this parameter

The rational behind this rule may be illustrated on an example. Assume that for a given parameter whose manufacturing tolerance is $[-\epsilon, \epsilon]$ the algorithm provides the result range $[a, b]$. If $b - a \geq 2\epsilon$ then we may choose as nominal value for the parameter any value in the range $[a + \epsilon, b - \epsilon]$: indeed to any such value we may add an arbitrary manufacturing tolerance in the range $[-\epsilon, \epsilon]$ with a result still in $[a, b]$. In other words the parameter value for the *real* robot will still be such that its representative points in the parameters space will belong to $\mathcal{Z}$.

Interval analysis-based method may be thought as a method to compute an *approximation* of the region $\mathcal{Z}$ in which the parts of $\mathcal{Z}$ that are too close to the border are eliminated.

We have now to explain how we may design an algorithm to calculate the region $\mathcal{Z}$.

4.1 Calculating $\mathcal{Z}$: an example

Up to now we have assumed that the performance requirement has a closed-form that can be interval evaluated. This is not always the case in robotics. For example assume that we consider the positioning accuracy $\mathbf{\Delta X}$ of the robot with respect to the joint measurement errors $\mathbf{\Delta\Theta}$. Both quantities are linearly related by

$$\mathbf{\Delta X} = J(\mathbf{X})\mathbf{\Delta\Theta}$$

where J is the Jacobian matrix of the robot, whose elements are functions of the pose $\mathbf{X}$ and of the design parameters.

The following requirement is classical: being given bounds $\mathbf{\Delta\Theta}^M$ on the joint errors determine the design parameters such that the robot's positioning errors are lower than given thresholds $\mathbf{\Delta X}^M$, whatever is the pose of the robot in a given workspace $\mathcal{W}$. Unfortunately for closed-chain robots the matrix J may be quite complex (or even may not be available) while its inverse J^{-1} may have a simple form. But it is possible to state the problem using only J^{-1}: find the design parameters $\mathcal{P}$ such that for all $\mathbf{X}$ in $\mathcal{W}$ all the solutions in $\mathbf{\Delta X}$ of the linear system $J^{-1}(\mathbf{X}, \mathcal{P})\mathbf{\Delta X} = \mathbf{\Delta\Theta}$ with $\mathbf{\Delta\Theta} \leq \mathbf{\Delta\Theta}^M$ are included in $\mathbf{\Delta X}^M$.

We have thus to solve a classical problem of interval analysis: being given an interval matrix $\mathbf{A}$ and an interval vector $\mathbf{b}$ determine an enclosure of all the solutions of the linear interval system $\mathbf{A}x = \mathbf{b}$ i.e. a region that includes the solution of $Ax = b$ for all A, b included in $\mathbf{A}, \mathbf{b}$ [15, 17]. It can be shown that classical methods of linear algebra (such as the Gauss elimination algorithm) may be extended to deal with this problem. We may directly use these methods to compute an enclosure of $\mathbf{\Delta X}$ and store as result the parameters boxes such that this enclosure is included in $\mathbf{\Delta X}^M$. But we may improve their efficiency: indeed these methods assume no dependency between the elements of $\mathbf{A}$ i.e.

the elements of the matrices A that are considered may have any arbitrary value within their ranges in $\mathbf{A}$. In our case there are dependencies between the elements of J^{-1} and not all possible values are allowed.

Our basic method is the Gauss elimination scheme. We compute an interval evaluation $\mathbf{A}^{(0)}$ of $\mathbf{A}$ and an interval evaluation $b^{(0)}$ of $\mathbf{b}$ (using the derivatives of the components of $\mathbf{A}, \mathbf{b}$ to improve these interval evaluations). The Gauss elimination scheme may be written as [17]

$$A_{ik}^{(j)} = A_{ik}^{(j-1)} - \frac{A_{ij}^{(j-1)} A_{jk}^{(j-1)}}{A_{jj}^{(j-1)}} \quad \forall\ i \text{ with } j > k \tag{1}$$

$$b_i^{(j)} = b_i^{(j-1)} - \frac{A_{ij}^{(j-1)} b_j^{(j-1)}}{A_{jj}^{(j-1)}} \tag{2}$$

The enclosure of the variable X_j can then be obtained from $X_{j+1}, \ldots, X_n$ by

$$X_j = (b_j^{(j-1)} - \sum_{k>j} A_{jk}^{(j-1)} X_k)/A_{jj}^{(j-1)} \tag{3}$$

We have improved the interval evaluation of the quantities appearing in the scheme by taking into account the derivatives of the elements of $A^{(0)}, b^{(0)}$ with respect to the pose and design parameters and propagating them by using the derivatives of the elements of $A^{(j-1)}$ to calculate the derivatives of the elements of $A^{(j)}$ and use them for the interval evaluation. Our experiments have shown that this lead to a drastic increase in term of the tightness of the enclosure.

Note also that this method may be used to determine what should be the design parameters so that any wrench in a set may be produced at any pose of $\mathcal{W}$ while the joint forces/torques are bounded. By duality the method can also solve the velocity problems (for bounded joint velocities find the design parameters such that any end-effector twist in a given set may be realized at any pose in $\mathcal{W}$).

4.2 A critical analysis of the zone calculation

We have presented in the previous section various methods to compute an approximation of the region $\mathcal{Z}$. However it is not possible to claim that we guarantee to get an approximation of the region that includes all possible values of the design parameters, up to the manufacturing tolerances, that will satisfy the performance index. Indeed for complex performances index the overestimation of interval arithmetics may be so large that only for very small boxes (i.e. whose width is lower than the manufacturing tolerances) we can guarantee that the performance index is satisfied. But the union of such small boxes, that may exist in the intersection of the $\mathcal{Z}_i$, may constitute boxes whose final width may be larger than the manufacturing tolerances.

Our experience however is that for robotics problem this is not the case. But a possibility to tackle this problem is to assume that the tolerances are much

lower than then real one. After calculating the approximation of the regions and their intersection we may then decrease the result by the real tolerances to get a safe design region.

4.3 Calculating the intersection of the $\mathcal{Z}$

As soon as an approximation of the regions $\mathcal{Z}_i$ have been determined as a set of boxes in the parameters space calculating their intersection is a classical problem of computational geometry with complexity $O(n \log n)$ for n boxes. But calculating the intersection may be avoided in a way that even speed-up the total calculation. Indeed assume that the region $\mathcal{Z}_1$ has been computed for the first requirement, leading to a list of boxes $\mathcal{L}^1$. For the second requirement instead of using $\mathcal{P}_0$ as single element of the list $\mathcal{L}_P$ (and thus looking for all parameters that satisfy the second requirement) we may use $\mathcal{L}^1$ as $\mathcal{L}_P$, thereby looking only for the parameters that satisfies both requirements. Proceeding along this line for all requirements will lead to a result that satisfy all requirements. A drawback however is that if one of the algorithm fail to provide design solutions (or if we want to modify a requirement) we may have to restart a large part of the calculation.

4.4 The algorithm in practice

As mentioned previously the algorithm are implemented in C++ using `BIAS/Profil` for interval arithmetics and our own interval analysis library `ALIAS`[1] that offer high-level modules that are combined for implementing the calculation of the region $\mathcal{Z}$.

4.5 Choosing the optimal design

Assume now that we have succeeded in computing the regions for all requirements and then their intersection $\mathcal{Z}_\cap = \cap \mathcal{Z}_i$. Clearly we cannot propose to the end-user an infinite set of solutions and our purpose is now to propose various design solutions whose representative points lie in $\mathcal{Z}_\cap$ (i.e. they satisfy the requirements). But a robot presents various performances, denoted secondary requirements, that may not be part of the main requirements but which can be used to help choosing the best design. Ideally the presented design solutions should be representative of various compromises between the secondary requirements. Unfortunately there is no known method to deal with this problem. Hence we just sample the region $\mathcal{Z}_\cap$ using a regular grid, compute the secondary requirements at the nodes of the grids and retain the most representative solutions.

Note that the algorithms for computing the region $\mathcal{Z}$ may also be used to verify that a given design (or a small family of design as, for example, the family of robots whose design parameters have values around nominal values and

[1] www.inria-sop.fr/coprin/logiciel/ALIAS/ALIAS.html

within manufacturing tolerances, called the *family of manufactured robot*) satisfy a requirement, in which case they will be much more faster. Using this property and as we will provide finally only a finite set of design solution we may relax the requirements when computing the regions. For example for the workspace algorithm instead of specifying a whole 6D region as desired workspace we may specify only a finite set of poses: this will allow a faster calculation of the region in the parameters space and we will only have to verify that the proposed final design solution indeed includes the whole 6 workspace.

Similarly it may happen that for a specific requirement an algorithm for computing the region $\mathcal{Z}$ is not available. But as soon as an algorithm for verifying the requirement for the family of manufactured robots is available our design method may still be applied.

4.6 Applications

As mentioned previously we have developed algorithms for computing the region $\mathcal{Z}$ for the following requirements: workspace, singularity detection, accuracy, velocity and static analysis. Such requirements are the most frequently encountered for practical applications. The design methodology has then been used for various practical applications: design of our own prototypes (for example the micro-robot MIPS for medical application [12]), fine positioning devices for the European Synchrotron Radiation Facility (ESRF) with a load over one tons and an absolute accuracy better than a micrometer [3], the CMW milling machine for high-speed manufacturing [21]. We are currently using this design methodology approach with Alcatel Space Industry for the development of an innovative deployable space telescope.

In each of these cases the on-the-shelf algorithms for calculating the region $\mathcal{Z}$ has to be adapted to deal with specificities of the application (for example the large workspace for the CMW milling machine implies that we have to deal with passive joint limits while the ESRF one, with a reduced workspace, such limits do not play a role). But the flexibility of interval analysis is large and has allowed us to solve the problem.

5 Conclusion

The proposed design methodology has the main advantages of providing a large panel of design solution with a guarantee on the satisfaction of the main requirements, even taking into account manufacturing tolerances. However its practical implementation needs some expertise in interval analysis for the algorithm to be efficient. A current restriction is that only non time-dependent requirements (i.e. requirements that are not solution of a differential equations) may be taken into account: for example we cannot deal with dynamics. However there is no theoretical impossibilities to deal with these requirements with interval analysis and this is a prospective for our work.

The development of this methodology has been guided by applications in very different domains: manufacturing, fine positioning, space and medical applications.

Finally the methodology has been developed to deal with robots and mechanisms design but may be extended to problems in other area as well.

References

[1] Boudreau R. and Gosselin C.M. The synthesis of planar parallel manipulators with a genetic algorithm. *ASME J. of Mechanical Design*, 121(4):533–537, December 1999.

[2] Chablat D., Wenger P., Majou F., and Merlet J-P. An interval analysis based study for the design and the comparison of three-degrees-of-freedom parallel kinematic machine. *Int. J. of Robotics Research*, 23(6):615–624, 2004.

[3] Comin F. Six degree-of-freedom scanning supports and manipulators based on parallel robots. *Review of Scientific Instruments*, 66(2):1665–1667, February 1995.

[4] Erdman A.G. *Modern Kinematics*. Wiley, New-York, 1993.

[5] Faraz A. and Payandeh S. A robotic case study: optimal design for laparoscopic positionning stands. In *IEEE Int. Conf. on Robotics and Automation*, pages 1553–1560, Albuquerque, April, 21-28, 1997.

[6] Gosselin C. *Kinematic analysis optimization and programming of parallel robotic manipulators*. Ph.D. Thesis, McGill University, Montréal, June, 15, 1988.

[7] Han C-S, Tesar D., and Traver A. The optimum design of a 6 dof fully parallel micromanipulator for enhanced robot accuracy. In *ASME Design Automation Conf.*, pages 357–363, Montréal, September, 17-20, 1989.

[8] Hao F. and Merlet J-P. Multi-criteria optimal design of parallel manipulators based on interval analysis. *Mechanism and Machine Theory*, 40(2), February 2005.

[9] Huang T., Jiang B., and Whitehouse D.J. Determination of the carriage stroke of 6-PSS parallel manipulators having the specific orientation capability in a prescribed workspace. In *IEEE Int. Conf. on Robotics and Automation*, pages 2382–2385, San Francisco, April, 24-28, 2000.

[10] Lee E., Mavroidis C., and Merlet J-P. Five precision points synthesis of spatial RRR manipulators using interval analysis. *ASME J. of Mechanical Design*, 126(5):842–849, September 2004.

[11] McCarthy J.M. Mechanism synthesis theory and the design of robots. In *IEEE Int. Conf. on Robotics and Automation*, pages 55–60, San Francisco, April, 24-28, 2000.

[12] Merlet J-P. Optimal design for the micro robot MIPS. In *IEEE Int. Conf. on Robotics and Automation*, Washington, May, 11-15, 2002.

[13] Merlet J-P. Designing a parallel manipulator for a specific workspace. *Int. J. of Robotics Research*, 16(4):545–556, August 1997.

[14] Merlet J-P. and Daney D. A formal-numerical approach to determine the presence of singularity within the workspace of a parallel robot. In F.C. Park C.C. Iurascu, editor, *Computational Kinematics*, pages 167–176. EJCK, Seoul, May, 20-22, 2001.

[15] Moore R.E. *Methods and Applications of Interval Analysis.* SIAM Studies in Applied Mathematics, 1979.

[16] Murray A.P., Pierrot F., Dauchez P., and McCarthy J.M. A planar quaternion approach to the kinematic synthesis of a parallel manipulator. *Robotica*, 15(4):361–365, July - August , 1997.

[17] Neumaier A. *Interval methods for systems of equations.* Cambridge University Press, 1990.

[18] Pittens K.H. and Podhorodeski R.P. A family of Stewart platforms with optimal dexterity. *J. of Robotic Systems*, 10(4):463–479, June 1993.

[19] Ryu J. and Cha J. Volumetric error analysis and architecture optimization for accuracy of HexaSlide type parallel manipulators. *Mechanism and Machine Theory*, 38(3):227–240, 2003.

[20] Stoughton R. and Arai T. A modified Stewart platform manipulator with improved dexterity. *IEEE Trans. on Robotics and Automation*, 9(2):166–173, April 1993.

[21] Wildenberg F. Calibration for hexapod CMW. In *2nd Chemnitzer Parallelkinematik Seminar*, pages 101–112, Chemnitz, April, 12-13, 2000.